Optical Tablature Recognition (OTR) System: Using Fourier Descriptors as a Recognition Tool

Lee Ling Wei , Qussay A. Salih , Ho Sooi Hock
The University of Nottingham Malaysia Campus
leelingwei@yahoo.co.uk , qussay@hotmail.com , Ho.Sooi-Hock@nottingham.edu.my

Abstract

This paper presents an optical recognition system for the guitar tablature. Images of guitar tablature are fed as input to the system whereby each image undergoes four main stages of processing to produce a music output in MIDI format. Algorithms both existing and self-devised were used. Each input image was first cropped to the desired region, followed by a process for removal of the string lines and detection of the numbers. Recognition of the numbers was carried out using Fourier Descriptors based on 8 selected feature points. Once completed, the numbers were matched to their corresponding chords and then rearranged and played. The algorithms and methods used within the system are presented here with a justification on the selection of Fourier Descriptors as the recognition tool.

1. Introduction

Optical Music Recognition (OMR) Systems have been the topic of research around the world and their main purpose is to convert images of paper-based music scores into digitised formats whereby the latter can be used to reproduce restored versions of faded or blurred scores, output the scores in a different format [1] or even transform the input into playable music files. These systems are capable of understanding and interpreting the given images like a human and producing the appropriate output. The focus of OMR systems has been on the accurate recognition of musical symbols or notes and ongoing research is still being carried out in this aspect.

The Optical Tablature Recognition (OTR) System presented here refers to existing techniques used in proposed OMR systems and by enhancing the adaptations of these ideas, a system is therefore constructed for the recognition of guitar tablature. Each guitar tablature consists of 6 lines, with possibly numbers falling on each line. These lines in fact represent the 6 strings on a guitar. An OTR system may prove useful for novice guitar players facing the problem of interpreting tablatures. When implemented on a larger scale basis, the system can be extended to aid musically inclined but visually impaired users in nurturing their talents.

2 types of guitar tablatures are commonly used. The first type has string lines running through the numbers while the second displays the numbers isolated from the strings. The system presented here is constructed using the first type (former) of tablature. Each tablature image undergoes 4 stages of processing, i.e. Pre-processing, Segmentation of Regions of Interest, Classification and Recognition, and Post-processing. The algorithms involved will be presented in Section 3 following reviews of currently used algorithms.

1.1. Template matching

One of the most frequently used methods for character recognition is template matching which requires a repository of samples for comparison. Matching for several characters may be efficient but to match a large array of input is time consuming. Considering the tablature numbers, there are a total of 10 possibilities – the digits 0 to 9. Each input image has to be matched against 10 separate templates, pixel by pixel, to find the best match. The sizes of the templates and the input are significant and rescaling has to be carried out whenever necessary to ensure that the best match is always obtained. However by using Fourier Descriptors, each input is assigned a set of descriptors which are compared to a 2-dimensional list of values. This is multiple of times faster than pixel matching.

1.2. Feature extraction and matching

Another method for character recognition is through the extraction of featured regions or points from the object in the image and then correlating them with a

known set of patterns. This technique can be faster and more efficient than template matching. However there exists a discrepancy if the image size is inconsistent. Different sizes of input will lead to varied distances between each feature point or region. This has an impact on the matching accuracy. Furthermore effective algorithms are needed to cleanly extract the targeted areas. Although feature points are required in Fourier Descriptors, the difference is that these points are taken at fixed intervals along the boundaries of objects instead of scanning for the areas of interest.

2. Background

A basic understanding of how a guitar is played and the method for reading the tablature are required in order to carry out the system development. A typical guitar –classical acoustic or electric – has 6 strings attached to its body as shown in Figure 1. Each of these strings is represented by a corresponding line in a tablature (Figure 2).

Figure 1. A portion of the guitar fretboard

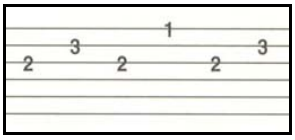

Figure 2. A portion of guitar tablature

For most tablature alignment, the topmost line corresponds to the topmost string on the fretboard and vice versa. The numbers indicate the positioning of the fingers – i.e. number 2 on the third string means holding down the third string on the second fret.

The font types and sizes used in guitar tablatures may differ from one publisher to another. For the purpose of this research, tablatures from the book "Classical for Guitar – Easy to Intermediate Classics and Transcriptions for Solo Guitar" arranged by Jerry Snyder and published by Alfred Publishing Co, Inc are used as illustrated in Figure 2.

As of current there is an OTR System for lute tablature [2] which is developed by Christoph Dalitz and Thomas Karsten from the Niederrhein University of Applied Sciences. Dalitz and Karsten have made use of the readily available **Gamera** framework which comprises a host of functions written in the Python language for the construction of their system. This OTR System employs the k nearest neighbour (*kNN*) method together with a grouping algorithm within **Gamera** for the recognition of the lute tablature symbols. A very high recognition rate was achieved through the combinatorial use of several features.

3. The system

The system uses images of type JPEG as input to ensure the system's capability to support the reading of low quality images format.

3.1. Pre-processing

The initial stage of the system comprises of pre-processing operations which enhances the quality of the input image. Firstly the image is converted to greyscale followed by a Power-Law Transformation [3] which can be represented by the following equation.

$$s = cr^{\gamma}$$

The values c and γ are positive constants. This equation is sometimes written as $s = (r + \varepsilon)^{\gamma}$ to account for cases of offset [3]. The Power-Law Transformation is a log transformation and fractional values of γ less than 0 classify dark pixel values into lighter regions while higher levels of γ map light regions into dark areas, therefore distinctly outlining the boundaries between objects and the background. Furthermore depending on the contents of the image, the Power-Law Transformation can also be used to eliminate noise.

Several values of γ have been chosen and each one tested against a sample input image. It was decided that a value of 2.0 was to be used. The following images show output samples using γ values of 1.0, 2.5 and 3.0 respectively. Fractional values produced entirely blank images and as such these values were not used.

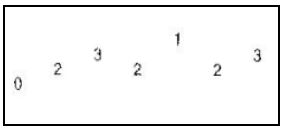

Figure 3(a). Power-Law transformation with = 1.0

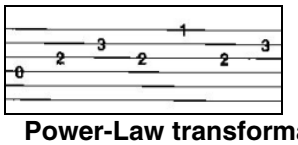

Figure 3(b). Power-Law transformation with = 2.5

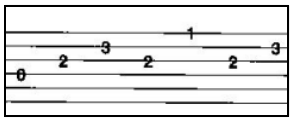

Figure 3(c). Power-Law transformation with = 3.0

A γ value of 2.0 produced the results as shown in Figure 4. Figure 4(a) contains Gaussian noise around the edges of the objects and by applying the Power-Law Transformation the image pixels are converted into binary values, therefore eliminating the noise blurring. This makes the process of thresholding easier as the pixel values around and on the edges are now consistent. However symptoms of 'jaggies' or the staircase effect may be noticeable after the application of this function. Nevertheless the binarised imaged will undergo several processes in the following stages which actually make use of these distinct edges.

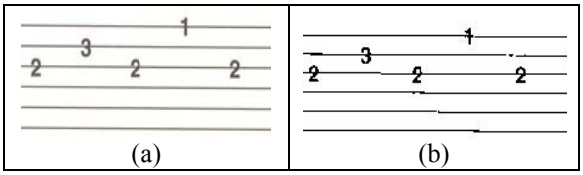

Figure 4. (a) Before Power-Law transformation. (b) After Power-Law transformation

3.2. Segmentation of regions of interest

The second stage is where the pre-processed image is reduced to the regions of interest. A complete guitar tablature consists of 2 sections as shown in Figure 5. The upper portion is made up of music notes transcribed for piano while the lower portion shows the tablature.

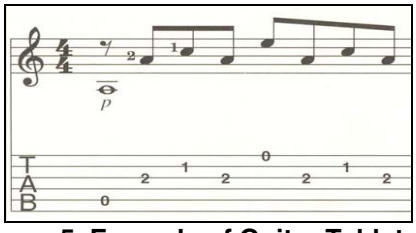

Figure 5. Example of Guitar Tablature

Several processes are executed on the image until the input image is reduced leaving only the tablature region. Firstly the music notes portion is cropped out of the image. This is done by making use of the standard features of both the tablature and the piano transcription manuscript. An algorithm was written to detect the leftmost vertical line which exists in both the manuscript and the tablature and which is parallel to each other. This line indicates the starting point for both the sections. A vertical scan is conducted starting from the left projecting from the top and when the first black pixel is hit, the scanning stops. The image is then cropped along the coordinates of the found black pixel and the resulting image is shown in Figure 6.

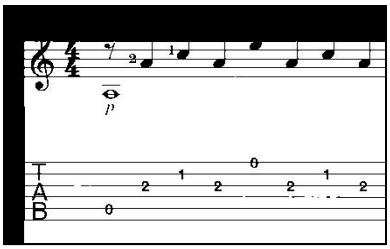

Figure 6. Cropping of areas of interest

The next step is the elimination of the manuscript leaving only the tablature region to process on. Some understanding of manuscripts is necessary here. A typical set of manuscript contains 5 lines and the distance between each line is fixed. On the other hand a guitar tablature has 6 lines as explained in earlier sections. This stage attempts to reduce the image until the tablature portion is left. This not only narrows down the scope for the recognition stage, it also saves processing time. To remove the unwanted section, an algorithm was designed to execute the cropping. The algorithm starts from the top of the image several pixels apart from the left edge and scans downward while keeping track of each line pixel it encounters. This process was carried out near to the edge as both the tablature and manuscript do not contain symbols that may interfere with the lines at this area. Whenever a black pixel is detected, the algorithm starts a counter for tracking all following white pixels until the next black line pixel is reached. The white pixel count is then temporarily stored into memory and the process is repeated again until the next white pixel count is obtained. These 2 sets of values are then compared against each other. Although the white distances between the lines may look the same to the naked eye, in fact there may be slight deviations between them in terms of image pixels. Therefore a small variance is introduced during the comparison of the 2 values.

To determine if the end of the manuscript section has been reached, the indicator is a large difference in the comparison of the white pixel counts. This is the stopping condition for the algorithm. Figure 7 shows the output with the manuscript portion eliminated.

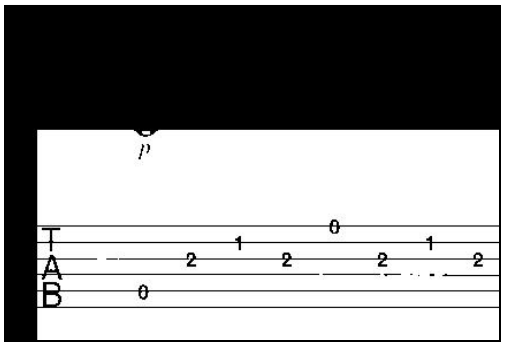

Figure 7. Elimination of the music manuscript section

Finally the string lines running through the numbers are removed and the necessary coordinate information are stored. During removal of the string lines the y-position of each line is kept into memory. Each line may be more than 1 pixel thick. A simple window is created to scan through the lines, tracking only the topmost black pixels within the window. These pixels give each line a distinct y-value. Occasionally a line may contain deviations (Figure 8) in parts of the line due to the Power-Law Transformation. This aspect is considered during the removal process so that a fairly clean output can be obtained which is important for the next and most important stage of the entire process. Figure 9 shows the output after the lines have been removed.

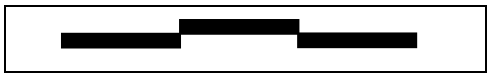

Figure 8. Deviation on line

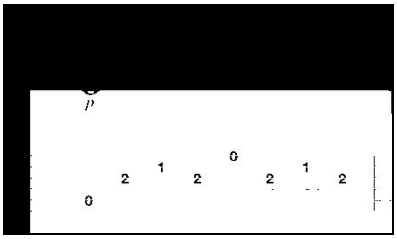

Figure 9. Removal of the string lines

The coordinate information or markers of each number is stored by running a 3x3 window through each line to detect the locations of the numbers. The y-values stored earlier were used as references for the horizontal scanning. There are 2 conditions involved here (refer to Figure 10 for the tagging of each window cell).

> Condition 1: if any of the diagonal neighbours (values v1, v3, v7, v9 in Figure 10) is black and the centre value (v5) is black, then a number exists in that region.
>
> Condition 2: if the vertical neighbours (v2, v8) are black and the centre value is black, then a number exists in that region.

v1	v2	v3
v4	v5	v6
v7	v8	v9

Figure 10. Detection window

These 2 conditions are applied based on analysis of the shapes of the numbers. Condition 1 is used to detect numbers 0-9 except 1 while condition 2 is used specially for the number 1. For each detected number the x-coordinate of the centre pixel of the window is stored. Using the window to scan through the numbers with the applied conditions ensure that an almost centre coordinate is obtained for each number. This is important when the numbers have to be cropped based on their markers. Following the location of the numbers, an inversion process was introduced to reverse the values of the image after which the numbers are extracted using the earlier gathered x and y markers. The extraction process introduces a small boundary around each number with the markers as referral points and each cropped number is then imposed onto another blank image. This procedure clearly eliminates unwanted noise in other areas of the image, leaving only the required information. Figure 11 shows a portion of an output ready for processing in the next stage.

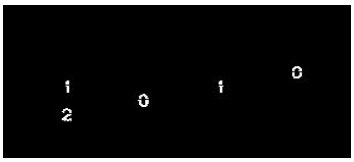

Figure 11. Image ready for recognition process

3.3. Classification and recognition

This stage is the core of the system and implements Fourier Descriptors as the recognition tool. Fourier Descriptors operate in the frequency domain and are good with shape classification given that a fixed set of data – in the case of image processing, a fixed set of points on the shape boundary – is given. The base of Fourier Descriptors is the Fourier Transform, and due to its nature of complex mathematics and imaginary numbers, Euler's formula is introduced to eliminate the aforementioned difficulties. Fourier Transform is represented by the following formula

$$F(u) = \int_{\infty}^{-\infty} f(x)\, e^{-i2\pi ux}\, dx$$

and its inverse is given by

$$f(x) = \int_{\infty}^{-\infty} F(u)\, e^{i2\pi ux}\, du$$

Both these equations form the Fourier Transform pair. In practice, the Discrete Fourier Transform is used due to the fact that the pixels of an image are never continuous [3]. Given N discrete samples, the equation thus becomes

$$a(u) = \frac{1}{K}\sum_{k=0}^{k-1} s(k) e^{-i2\pi uk/K}, \text{ u = 0, 1, 2,…k-1}$$

Euler's formula is represented by

$$e^{j\theta} = \cos\theta + j\sin\theta,\ \theta = -2\pi uk/K$$

which is introduced to convert the exponential value in Fourier Transform into the sine and cosine domains. Substituting Euler's Formula into the Discrete Fourier Transform, the following proof is obtained.

$$a(u) = \frac{1}{K}\sum_{k=0}^{k-1} s(k)[\cos(-2\pi uk / K) + j\sin(-2\pi uk / K)]$$

By assuming that P = -2πuk/K and that *s(k)* is the x and y coordinates of a particular pixel represented in complex number form *x+jy*, the equation then becomes

$$a(u) = \frac{1}{K}\sum_{k=0}^{k-1} [x_k \cos P - jx_k \sin P + jy_k \sin P + y_k \sin P]$$

The above equation contains both the real and imaginary portions. Separating them reveals 2 equations we can use

$$a(u) = \frac{1}{K}\sum_{k=0}^{k-1} x_k \cos P + y_k \sin P \qquad \ldots(1)$$

$$a(u) = \frac{1}{K}\sum_{k=0}^{k-1} jy_k \cos P - jx_k \sin P \qquad \ldots(2)$$

Equation 1, which is the x component, represents the summation of the real values while Equation 2 represents the summation of the imaginary values, the y component. Together there 2 values form a Fourier Descriptor pair for a point, or otherwise more commonly known as a descriptor.

In the above equations a(0) is the geometrical centroid of a shape and the result of this descriptor alone is a circle situated about the centre of the shape [4]. The low frequency descriptors give the gross boundary of the shape while the high frequency ones describe the shape in detail. By using a set of computed Fourier Descriptors for a known object, a new form of matching based on calculated values can be obtained. Every set of Fourier Descriptors can be used to rebuild the original shape – this attribute makes it especially useful to track shape information, making Fourier Descriptors a versatile tool.

Fourier Descriptors are able to overcome weak discrimination abilities of moment and global descriptors as well as noise sensitivity in shape signature representations [5]. Moment descriptors may describe information about an object in overall, however the higher order invariants are difficult to achieve. Global descriptors are generally simpler but to obtain a good result requires that the shapes compared contain a unique and distinct feature that separates one from another.

Shape recognition and classification using Fourier Descriptors can be conducted by comparing all the components except the first within the set of descriptors to a set of default and tested values. During the matching process, the unknown object will be classified to the sample that has the highest number of matches to it. By using the coordinate markers gathered from the previous stage as references, a normalisation process is carried out to ensure that the coordinates of each number stay within a specified range regardless of the position of the number on the input image. This step is important as it helps keep the consistency of the values used for the calculation of the Fourier Descriptors.

The normalisation process works by reducing each number's region built from the coordinate markers until the edges of the region hit the edges of the number. Then the coordinate values of the leftmost and topmost point are tracked with the remaining coordinates subtracted from these initial values. This converts the region values into a fixed set of range starting from 0. After this process, the next step is to extract feature points from each number's boundary for calculation of the descriptors. Many methods are available for this process and one of the most frequently used functions is the boundary tracing algorithm. However this algorithm is difficult to implement as it consumes time and has a high overhead. Furthermore the boundary shapes of 2 numbers may not necessarily be the same to each other in terms of their pixel arrangements. Due to the high number of pixels obtained from boundary tracing, the speed and time factors for calculation of a single number input are thus affected. Since Fourier Descriptors are able to work with any number of points and are sensitive to the information gathered from these input points, a measure was taken to extract 8 feature points from each number character. These 8 points are taken from specific locations using a derived algorithm. The locations chosen are based on analysis – these are the regions where the pixels, when extracted, describe the general shape of the number uniquely. A combination of linear equations is used in the extraction of the 8 feature points. Figure 12 shows the structure for the linear equations and the positions where the points are taken.

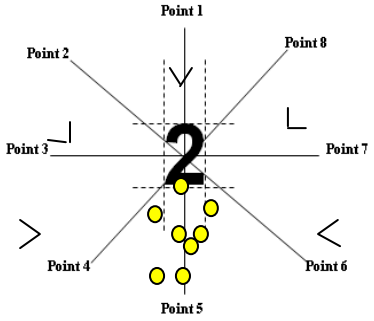

Figure 12. The linear equations cutting through the number

Table 1. Implementation of Fourier Descriptors

```
for u=0 to num_of_points
      for v=0 to num_of_points

            p = (2*PI*u*v)/num_of_points

            normalised_x = current_x – start_x
            normalised_y = current_y – start_y

            real[u] += normalised_x * cos(p) +
                        normalised_y * sin(p)
            imaginary[u] += normalised_y * cos(p)
                              –normalised_x *
sin(p)

      end for

      real[u] /= num_of_points
      imaginary[u] /= num_of_points

end for
```

Based on the Figure, it can be seen that the lines all cross through the centre of the number. Each of these lines starts projecting from the outside and when a black pixel is hit - which is a pixel on the boundary of the number, then the searching function halts. The extraction of the 8 points is done in an anti-clockwise manner. The direction and order the points are taken are vital as Fourier Descriptors depend on the order of the input data as well.

Upon extraction and storing of the feature points, the data is then sent to the next function which uses these points to calculate the descriptors. An implementation of the equations behind Fourier Descriptors is shown as below. This implementation is adapted from Yerin Yoo's C pseudocode [4] with added normalisation.

The normalisation process is first executed before the data is used for calculation of the Fourier Descriptors. Each number will have 8 pairs of descriptors due to the 8 points extracted. These 8 pairs of components or each Fourier Descriptor set is then compared against a databank of default and checked values for classification of the numbers. Once all the numbers have been grouped to their respective clusters, they are sent to the next stage together with their positioning information for rearrangement and post-processing.

3.4. Post-processing

The final stage of the process involves assigning the appropriate chords to the recognised numbers, inserting the correct rhythm as well as presenting the output in MIDI format. First of all the numbers are grouped together with their respective string lines to determine the output chord. As mentioned earlier, the same number – when located on different string lines, produces different chords. Number '2' on string 5 produces chord B while the same number on string 3 produces chord A. Therefore a function is written to pair up each string-number combination with their corresponding chords. Once this step is completed, the positions of the numbers are rearranged using the Quicksort algorithm. To the human reader, the mind can easily perceive the flow of the numbers along their lines and from left to right but machines are incapable of doing so. The algorithm sorts the numbers using their x coordinates into the order as seen in the tablature. 2 or more numbers that may be aligned vertically together (Figure 13) may not give the exact same x reference coordinates. However it can be said that the distance between them will be very small. As such there is a function that carries out subtraction between the coordinates and a checking statement is implemented such that numbers with a distance of 10 pixels of less between them will be grouped together.

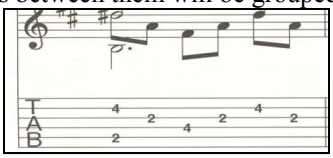

Figure 13. Boxed Area Showing Numbers Aligned Vertically

At the same time, this function approximates the duration of the current chord before the next chord is played. In a musical piece, chords are played against a specified tempo and each chord may have its own distinct duration. This variation is what makes a musical piece or song soothing to the ear. To achieve this, the distance between every 2 numbers on a string

has been used as the duration setting factor. This involves some simple calculations – for example, assuming the distance between 2 numbers is 100 pixels, and by using a reference of 200 pixels for duration of 2 seconds, we have a duration of 1 second for the first number chord. When all the chords have been arranged in order, they are then compiled and sent to a MIDI sequencer. 6 tracks are initialized – one to represent each string. Each track is then filled with the chords of the string it represents together with their calculated durations. The sequencer then takes all these tracks, compiles them and plays the synthesized output.

4. Results

The system was tested against 5 images for recognition accuracy whereby each of the images has a size of approximately 1170 by 250 pixels. The following tables show the results obtained for each image.

Image 1 : Allegro I	
Total numbers in image	24
Recognition successes	21
Recognition failures	3
Recognition rate	**87.50%**

Image 2 : Allegro II	
Total numbers in image	24
Recognition successes	20
Recognition failures	4
Recognition rate	**83.33%**

Image 3 : Study A II	
Total numbers in image	36
Recognition successes	32
Recognition failures	4
Recognition rate	**88.89%**

Image 4 : Waltz I	
Total numbers in image	15
Recognition successes	13
Recognition failures	2
Recognition rate	**86.67%**

Image 5 : Study B IV	
Total numbers in image	21
Recognition successes	19
Recognition failures	2
Recognition rate	**90.48%**

Summary of results:

	Test Image	**Total**	**Successful**	**Recogniti**

			Recognition	**on Rate (%)**
1.	Allegro I	24	21	87.50
2.	Allegro II	24	20	83.33
3.	Study A II	36	32	88.89
4.	Waltz I	15	13	86.67
5.	Study B IV	21	19	90.48
			Overall Percentage	**87.50**

Out of a total of 120 numbers involved in the recognition, 105 numbers have been successfully recognised with the overall recognition rate achieved at 87.5%. One of the reasons for recognition to fail could be due to differences in the edges of the numbers. As mentioned earlier Fourier Descriptors are sensitive to changes and a change in any point may lead to a different set of descriptors being calculated for that number resulting in the number being classified wrongly. Another cause for failure may be due to the quality of the input images used. If the input image has been scanned tilted, the processed image may contain elements that interfere with the program. Although Fourier Descriptors are able to handle rotation, however this has not been implemented in the project as the processing consumes time. The result is a less than efficient elimination of the string lines, as can be seen in Figure 14. These 'leftover' string lines can lead to inaccurate calculation of the descriptors as the required points may have been extracted at the wrong places.

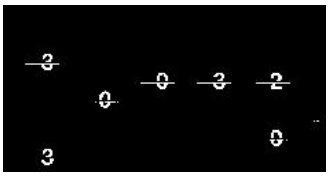

Figure 14. Leftover String Lines in an Image

5. Discussion

In general the recognition rate using Fourier Descriptors is relatively high considering that only 8 feature points were used. If the number of feature points were to be increased, chances are that the recognition rate will improve as the overall failure percentage will drop lower. At the same time, if the mentioned difficulties could be overcome, Fourier Descriptors would yield good results for character recognition as well as shape classification. Compared to other methods like template matching or the use of neural networks, Fourier Descriptors are faster as it involves only mathematical calculations. As mentioned, template matching requires a databank of images while neural networks have to be trained until

the networks become stable and consistent to produce the best results.

The Optical Tablature Recognition (OTR) System is relatively new to the world of music although this concept has been around for some time in the form of Optical Character Recognition and Optical Music Recognition systems. However to obtain a fully accurate recognition system requires more than just image processing techniques. Input images – whether scanned or captured, should have their quality preserved as much as possible. A solution to maintain the quality is to obtain the input images in raw format from the capturing devices themselves and carry out image processing algorithms on them. This step prevents Gaussian noise from interfering with the pixels of the image, giving a more accurate output.

6. Conclusion

The Optical Tablature Recognition (OTR) System takes in images of guitar tablature, processes them and presents the output in the form of MIDI music files. The main tool used for recognition is Fourier Descriptors which work in the frequency domain and is a good method for shape classification. Application of Fourier Descriptors in the recognition of tablature numbers using a small number of feature points yielded a reasonably good result with an overall recognition success rate of 87.5%. The OTR system, when implemented on a large-scale basis, may prove to be beneficial to users who are musically inclined but visually impaired.

7. References

[1] Ng, K., "Music Imaging: Optical Music Recognition and Restoration", University of Leeds.

[2] Dalitz, C., Karsten, K., "Using The Gamera Framework For Building A Lute Tablature Recognition System", Proceedings of the Sixth International Conference on Music Information Retreival, 2005, pp. 478-481.

[3] Gonzalez, R.C., Woods, R.E., *Digital Image Processing, Second Edition*, Pearson International Education, 2002.

[4] Yoo, Y., "Tutorial on Fourier Theory", March 2001.

[5] Zhang, D., Lu, G., "A Comparative Study on Shape Retrival Using Fourier Descriptors with Different Shape Signatures", Monash University.

[6] Choudhury, G.S., DiLauro, T., Droettboom, M., Fujinaga, I., Harrington, B., Macmillan, K., "Optical Music Recognition System within a Large-Scale Digitization Project", John Hopkins University.

[7] Ng, K., "Optical Music Recognition: Stroke Tracing and Reconstruction of Handwritten Manuscripts", University of Leeds.

[8] Folkers, A., Samet, H., "Content-based Image Retrieval Using Fourier Descriptors on a Logo Database", Medical University of Lubeck, University of Maryland at College Park.

[9] Abraham, B., Camps, O.I., Sznaier, M., "Dynamic Texture with Fourier Descriptors", The Pennsylvania State University.